SERBIAN JOURNAL OF ELECTRICAL ENGINEERING
Vol. 20, No. 3, October 2023, 391-405

UDC: 004.451.57:621.391.6 **DOI: https://doi.org/10.2298/SJEE2303391L**

Impact of Buffering Time Reduction on False Congestion Detection in TCP Vegas Over OBS Networks

Van Hoa Le[1]

Abstract: Since Transmission Control Protocol (TCP) is the dominant protocol on the Internet and optical burst switching (OBS) networks represent the optical transmission solution that can meet future high bandwidth requirements, TCP and OBS networks represent a possible combination for the next-generation Internet. However, a problem with this combination model is that operations at the OBS layer need to be controlled so that false congestion detection at the TCP layer is minimised. Considering the combination of TCP Vegas and OBS, congestion detection at the TCP layer is based on the round-trip time (*RTT*), where congestion is detected if the current *RTT* exceeds a given RTT_{max}. The extra delay may be due to some actual loss, but it may also be due to the extended execution time caused by some OBS layer operations. Therefore, reducing the time of operations at the OBS layer will reduce false congestion detection at the TCP layer. This paper investigates the impact of reducing the buffering time at the ingress node on the false congestion detection rate at the TCP layer. The reduction of the burst buffering time is accomplished by nesting the offset time in the assembly time. The simulation results and analysis show that the transmission efficiency in TCP Vegas over OBS networks is significantly improved; the throughput remains high, and the delivery success rate is significantly increased compared to the conventional buffering mode.

Keywords: TCP Vegas, OBS, False congestion detection, Buffering time reduction, Improved hybrid assembly.

1 Introduction

With increasing Internet traffic growth and recent advancements in wavelength division channel (WDM) technology, WDM networks seem to be the most suitable choice for backbone infrastructures, allowing explosive bandwidth requirements to be met. Optical fibre offers advantages such as a low signal attenuation, very low bit error rate and small signal distortion and also promises a huge bandwidth potential thanks to advancements in WDM technology [1, 2].

[1]Hue University, Hue, Vietnam; E-mail: levanhoa@hueuni.edu.vn

The development of optical networks is associated with the introduction of optical switching techniques. Up to now, three models of optical switching have been proposed: optical packet switching (OPS), optical circuit switching (OCS) and optical burst switching (OBS) [3, 4]. In the OCS model, an end-to-end optical channel is established between a source-destination node pair such that no optical-electrical-optical (OEO) conversion takes place at the intermediate nodes. Setting up this dedicated optical channel leads to the inefficient use of bandwidth because Internet traffic is often volatile. With OPS networks, this waste of bandwidth can be avoided by encapsulating Internet Protocol (IP) packets coming from various sources in the same optical packet and then switching directly in the optical domain. Switching synchronised and fixed-length optical packets minimises contention, but this is practically impossible. Furthermore, the absence of optical buffers and high-speed switches makes the OPS model unfeasible [3]. To overcome these issues and still achieve switching in the optical domain, the OBS network is therefore proposed.

OBS is considered a combination of OCS and OPS [4]. In OBS networks, the ingress node is responsible for aggregating data from the upper layer, which could be a Transmission Control Protocol (TCP) layer, into blocks called data bursts (or bursts). A burst control packet (BCP) is generated and sent ahead of time to configure the switches at the intermediate (core) nodes. Then, the burst is transmitted into the network and optically switched at these core nodes. The establishment of a path for a burst can be fixed; a BCP reserves resources and releases them after the burst transmission is completed [5]. At the egress node, the reverse process is performed. The original packets are extracted from the burst and forwarded to the upper layer.

OBS networks may represent a future Internet infrastructure in the physical layer, while TCP is still the dominant protocol in the network layer today and will be in the future. Therefore, the combination of TCP and OBS could be a viable solution for the next generation of optical Internet. However, a problem in this combination model is that the TCP congestion control mechanism needs to be able to detect false congestion caused by operations at the OBS layer. Specifically, considering the combination of TCP Vegas and OBS, congestion detection at the TCP layer is based on the round-trip time (*RTT*); congestion is detected if the current *RTT* exceeds a given RTT_{max}. Thus, when there is an increase in the amount of time needed to perform operations such as buffering, retransmission or deflection routing at the OBS layer, TCP Vegas will reduce the congestion window size, and this will result in reduced throughput in TCP over OBS networks.

The transmission delay of IP packets over an OBS network is made up of the total buffering time (including the assembly time and offset time) at the ingress node, the propagation time over the network, the switching time at the

intermediate nodes and the disassembly time at the egress node. For an end-to-end connection over a given OBS network, the propagation time, switching time and disassembly time are irreducible Therefore, only the buffering time can be reduced. Some studies have proposed solutions for reducing the buffering time that nest the offset time in the assembly time, [6 – 10]. This approach has been proven to be effective in reducing the transmission over OBS networks [10]. In this paper, we study the impact of buffering time reduction on the false congestion detection rate for TCP Vegas over OBS networks. The main contributions of the paper include the following:

– Previous TCP Vegas over OBS models are analysed and their drawbacks are pointed out.
– The impact of buffering time reduction on false congestion detection is evaluated using the throughput entering the OBS networks.

The next part of the paper is organized as follows: Part 2 presents basic information about TCP Vegas and bust formation. A review and evaluation of models related to TCP Vegas over OBS is provided in Section 3. The reduction of the buffering time by nesting the offset time in the assembly time is illustrated in Section 4. An analysis of the simulation results is described in Section 5 and Section 6 is the conclusion.

2 Background

2.1 TCP Vegas

TCP Vegas is a variant of the TCP protocol that controls the current congestion window size (*cwnd*) based on throughput values instead of packet loss like TCP Reno [11]. TCP Vegas measures the *RTT* value of each packet transmission to estimate available bandwidth and congestion in the network. TCP Vegas first calculates RTT_{base}, which is the minimum measured *RTT*, to estimate the transmission delay as well as the queuing delay. Then, TCP Vegas calculates the expected throughput (*Expected*) according to (1) [12]:

$$Expected = \frac{cwnd}{RTT_{base}}, \tag{1}$$

Next, TCP Vegas will calculate the actual throughput (*Actual*). For each *RTT*, TCP Vegas calculates *Actual* using the most recently measured *RTT* with (2) [12]:

$$Actual = \frac{cwnd}{RTT}. \tag{2}$$

TCP Vegas then compares *Actual* and *Expected* and calculates the difference (*Diff*) as follows:

$$Diff = Expected - Actual\ . \tag{3}$$

TCP Vegas defines two threshold values for controlling *Diff*, which are α and β. The behaviour of the congestion window is summarised as follows:

$$cwnd = \begin{cases} cwnd + 1, & Diff < \alpha, \\ cwnd, & \alpha \le Diff \le \beta, \\ cwnd - 1, & Diff > \beta. \end{cases} \tag{4}$$

If *Diff* < α, then TCP Vegas will increase the window size linearly for the next *RTT*. If *Diff* > β, TCP Vegas linearly reduces the congestion window size for the next *RTT*. Otherwise, TCP Vegas leaves the window size unchanged.

If *Actual* is much less than *Expected*, TCP Vegas assumes that there is a possibility that the network is congested, so the traffic rate is reduced. Otherwise, if *Actual* is too close to *Expected*, TCP Vegas assumes that the connection may not be fully utilised and thus increases the traffic rate.

2.2 Burst formation

The formation of a burst consists of several stages. Initially, IP packets coming from the TCP layer are classified according to their destination and quality of service. These packets are then temporarily stored in the assembly queue corresponding to the above classification. When an assembly threshold is reached, the IP packets in the queue are aggregated into the same burst. A BCP is sent ahead to reserve the resource and the burst is sent after a certain offset time.

A burst is formed usually based on three main methods as follows:

1. Timer-based algorithms [13]: In a timer-based assembly algorithm, a burst is formed and sent into the network at regular intervals. Therefore, the burst length can vary depending on the incoming load. The timer should be chosen carefully, since certain operations are affected by this value, including congestion window control at the TCP layer. If the timer chosen is too long, the delay suffered by a packet may not be acceptable. However, if the selected timer is too short, a large number of BCPs and bursts are sent into the network.
2. Length-based algorithms [13]: In a length-based assembly algorithm, a threshold, which specifies the maximum number of packets in each burst (B_{max}), is used. It begins counting when the first packet arrives and continues counting until the total number of packets in the queue exceeds the threshold. A minimum threshold (B_{min}) can also be used to prevent the formation of bursts that are too small in size, resulting in poor bandwidth utilisation, and are not suitable for existing burst switches. Once the burst is formed, it is sent into the network.

3. Hybrid algorithms [13]: This type of algorithm is a combination of timer-based algorithms and length-based algorithms; the aim is to take advantage of each algorithm and merge them into a single algorithm. With the hybrid algorithm, a burst is formed when the timer specifies that a burst should be formed or when the length threshold is reached (whichever occurs first). A burst is sent into the network if it reaches the maximum size B_{max} or if the timer specifies that a burst should be formed. The condition that the burst length must be greater than B_{min} also needs to be met. If the burst has not reached the minimum size, padding [14] is an option that can be used to ensure that the burst length is greater than B_{min}.

3 Models Related to TCP Vegas Over OBS

In the TCP over OBS model, the delay of operations at the OBS layer often has an impact on false congestion detection at the OBS layer. There have been many studies focused on improving the congestion control model of TCP Vegas to reduce the false congestion detection rate. The following is an analysis of these related models.

Leon and Sallent [15] performed a simulation-based study that examined the impact of burst assembly on the TCP Vegas throughput. The simulation results show that, if the timer is short, TCP Vegas behaves as if no burst assembly took place, but when the timer is longer, *cwnd* value stabilises away from *Expected* value due to the impact of burst assembly.

Ramantas et al. [16] performed a study on different variants of TCP with NS-2. The results show that the throughput of TCP Vegas and TCP New Vegas [17] decreases compared to FAST TCP [18] when the timer is changed from 1.0 ms to 0.001 ms. Among protocols that consider end-to-end latency as a measure of congestion, FAST TCP appears to be more consistent in its delivery success rates, when compared to TCP Vegas and TCP New Vegas for both length threshold-based assembly and timer-based assembly protocols.

Poorzare et al. [13] proposed an alternative method for choosing the best timer to generate bursts that have little impact on TCP Vegas. Through analysis and simulation, these authors showed that setting the timer using different contention probabilities and getting closer to a probability of 0.1 improved the performance. Simulation results in NS-2 were used to prove this theory. Another extension of TCP Vegas by Poorzare et al. [20] combines both the assembly based on the maximum size threshold and the assembly based on a timer. This paper attempted to strike a balance between the maximum burst size and the corresponding assembly time with different contention probabilities to improve the performance of TCP Vegas over OBS networks. The results from the simulation on NS-2 also demonstrated the effectiveness of this extension.

Shihada et al. [21] proposed a threshold-based TCP Vegas that can differentiate between increased *RTT* due to network congestion and increased RTT due to retransmission or misrouting in light-loaded OBS networks. The improved TCP Vegas algorithm in [21] can be used to more accurately detect congestion in the OBS network. This is achieved with continuous retransmission or offset routing, thereby improving throughput compared with the original version of TCP Vegas. Shihada et al. [22] provided an analysis of the throughput performance of a threshold-based TCP Vegas source for OBS networks with burst retransmissions. According to the analytical model, the authors obtained the steady-state TCP throughput and observed an appropriate threshold value, leading to the optimal throughput. The analytical model provides insight into the impact of contention and burst loss on TCP throughput.

TCP Vegas works by evaluating the *RTT* delay of packets, while the additional delay caused by packet loss in bufferless OBS networks always occurs due to congestion or contention. TCP Vegas cannot distinguish between the delay caused by congestion and the delay caused by contention, so there will flaws in the decision-making of the congestion window control method. Poorzare and Jamali [23] proposed a fuzzy-based solution that can distinguish between burst drops due to congestion and burst drops due to contention. To that end, Jamali [23] used both *RTT* and RTT_{base} to estimate the network congestion. As a result, the performance of the OBS network is improved by controlling congestion using fuzzy logic. An improved algorithm of Poorzare and Abedidarabad [24] uses fuzzy logic and some threshold to divide the network into several regions. This improved algorithm can better distinguish between burst drops due to congestion and burst drops due to contention in the network. The simulation results show that the improved algorithm outperforms TCP Vegas in terms of the throughput and delivery success rate.

Abedidarabad and Poorzare [25] proposed a new system based on a probabilistic method that can distinguish between sudden drops due to congestion and sudden drops due to collision to improve network performance. Poorzare et al [26] proposed a new algorithm called AVGR (average of RTT) based on several mathematical equations to prevent TCP degradation. It calculates the average of RTT over three different time periods. Then, based on the results obtained, the congestion window control mechanism will be adjusted. The results show that AVGR performs better than TCP Vegas in terms of throughput capacity. In addition, the average sending rate of the AVGR algorithm is higher than that of the TCP Vegas algorithm, and its fluctuations are also significantly smaller than that of the TCP Vegas algorithm.

In summary, the above approach considers the OBS layer as a black box and only tries to analyse the behaviour of the traffic to make appropriate adjustments to the congestion window. This paper takes the opposite approach: OBS layer

operations are controlled so that their execution times have as little impact as possible on the delay sensitivity of TCP Vegas. Specifically, this paper investigates the impact of reducing the buffering time on the false congestion detection rate in TCP Vegas over OBS networks.

4 Buffering Time Reduction Method

The transmission delay of IP packets over an OBS network can come from various operations, including buffering at the ingress node, propagation within the core network, processing and switching at the core nodes and disassembly at the egress node (Fig. 1). For a particular connection within a given network, the propagation time, switching time, and disassembly time are irreducible, so the transmission time can only be reduced by reducing the buffering time.

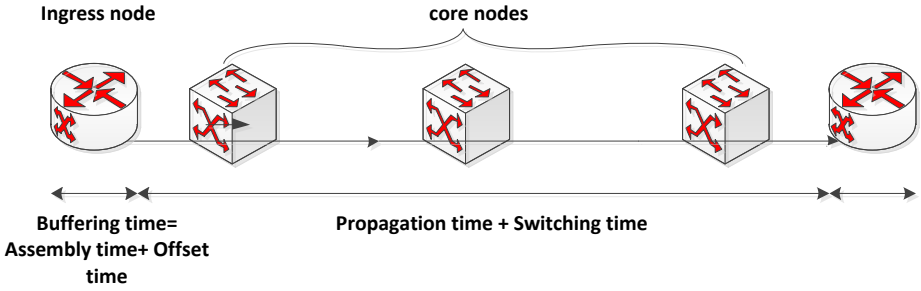

Fig. 1 – *The transmission delay includes the buffering time, propagation time, switching time and disassembly time.*

In the basic model, the buffering time at the ingress node includes its assembly time and offset time, as shown in Fig. 2a. In order to reduce the buffering time, some studies have proposed nesting the sliding offset time in the assembly time [6–10]. As depicted in Fig. 2, the offset time can be partially (Fig. 2b) or fully (Fig. 2c) nested in the assembly time. Nesting the entire offset time in the assembly time will reduce the buffering time the most, i.e. it will be reduced by one offset time. However, since the burst length value must be carried in the BCP and the burst is incomplete at the time that the BCP is sent, no actual burst length value is available. The solution used is to predict the burst length at the time that the BCP is sent [6–10].

The prediction needs to be accurate because if the actual length is longer than the predicted length, then resource reservation at the switches will be insufficient and the switching will cause data loss; on the other hand, if the actual length is shorter than the predicted length, the reserved resources will be redundant and cause waste. Among the proposed buffering time reduction models, the model in [10] has the best predictive ability. In addition, the assembly mechanism in [10]

dynamically adjusts the length threshold according to the predicted length, thus maximising the prediction accuracy and making more efficient use of the reserved resources.

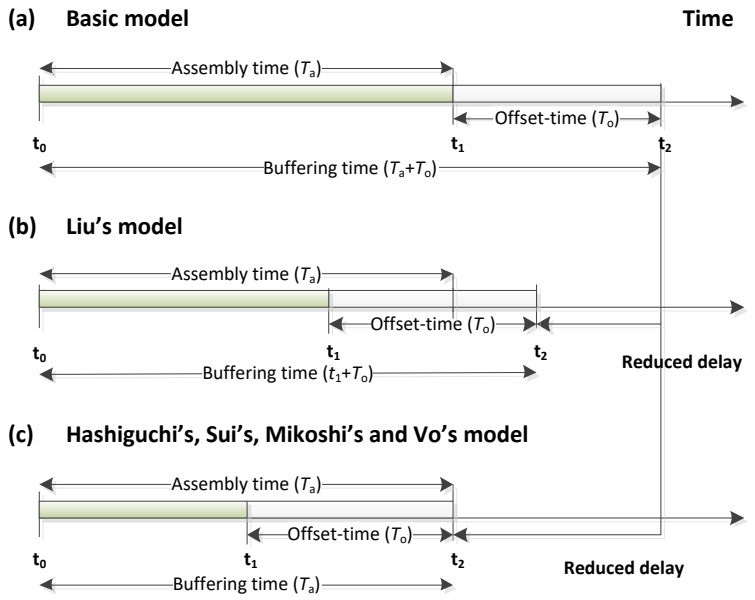

Fig. 2 – *The buffering time is reduced by sliding the offset time into the assembly time.*

This paper exploits the buffering time reduction method presented in [10], the Optimal Burst Assembly for Delay Reduction (OBADR) method, to examine the impact of buffering time reduction at the ingress OBS node on false congestion detection at the TCP Vegas layer. The model of TCP Vegas over OBS with OBADR is described as follows.

An IP packet coming from the TCP Vegas layer is queued at the OBS layer's ingress node. Depending on the destination and service class, the IP packet is buffered in the corresponding queue. The queue timer is turned on (at time t_0) when the first packet arrives at the queue (Fig. 3). At time t_1, a BCP is formed. The BCP carries the burst length value that is estimated based on the number of IP packets currently buffered in the queue and the previous packet arrival rate according to (5) [10]:

$$L_e = L_w + T_o \lambda_e , \tag{5}$$

where L_w is the length measured in the period $T_a - T_o$ and λ_e is given by (6):

$$\lambda_e = (1-\alpha)\lambda_{avg} + \alpha \frac{N}{T_a - T_o} , \tag{6}$$

where α is a weight factor, λ_{avg} is the average rate of packets that arrived previously, N is the number of packets currently being buffered and α is equal to $\lambda_{cur}/(\lambda_{avg} + \lambda_{cur})$.

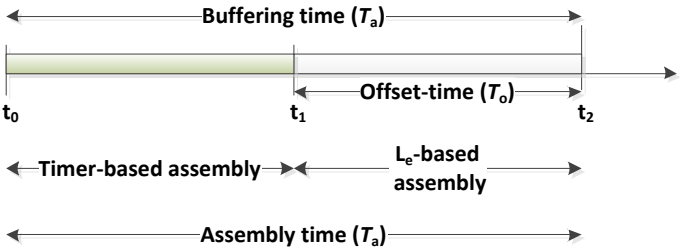

Fig. 3 – *The buffering time of packets with OBADR.*

OBADR uses an improved hybrid assembly method in which the aggregation is based on the timer at time t_1 and the burst is formed when the queue length reaches the threshold L_e. This ensures that the burst length value in the BCP is always exactly equal to the actual formed burst length. This is the advantage of the model consisting of TCP Vegas over OBS with OBADR; it not only reduces the buffering time but also reduces the data loss inside the OBS network. This reduces the extra delay in the *RTT* of packets and thus reduces the false congestion detection rate at the Vegas TCP layer.

5 Evaluation of the Impact of Buffering Time Reduction on False Congestion Detection

Simulations are performed on the NS-2 simulator [27] with the package obs-0.9a. The simulated network is a National Science Foundation Network (NSFNET) (Fig. 4); the bandwidth per link is 10 GB/s, and there are eight channels per link, including two control channels and six data channels. The simulation time is 10 s. Packets arriving at the ingress OBS node queue have a Poisson distribution. When congestion is detected, TCP Vegas reduces its congestion window, and as a result, the throughput into the OBS network decreases. Therefore, evaluating the impact of buffering time reduction on false congestion detection can be done by evaluating the impact of buffering time reduction on the throughput into the OBS network.

Therefore, the simulation objectives are the following:

- evaluating the throughput achieved when the buffering time changes (mainly the timer of burst assembly);
- determining the impact of buffering time reduction on false congestion detection using the throughput entering the OBS network. The byte loss rate and the delivery success rate are also considered in this case.

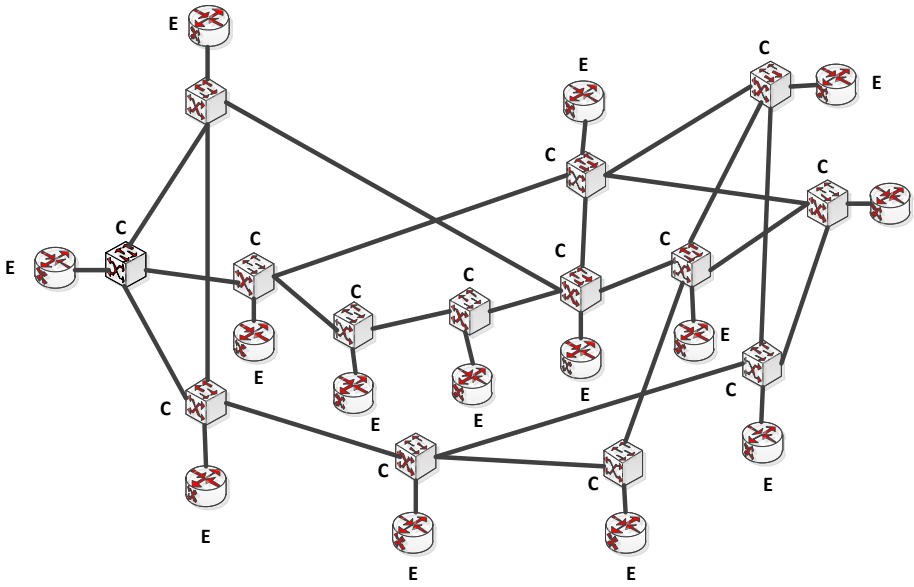

Fig. 4 – *The simulated network.*

When the timer is increased from 10 μs to 500 μs the throughput entering the network decreases (Fig. 5). The reason for this is that when the timer is increased, the waiting time of IP packets at the ingress node increases, thus increasing the transmission delay over the OBS network. As a result, some *RTT*s exceed RTT_{max}, and TCP Vegas reduces the congestion window because it assumes that congestion has occurred. In fact, this is not always the case because the data loss rate (Fig. 6) is reduced and the delivery success rate (Fig. 7) is increased at the OBS layer.

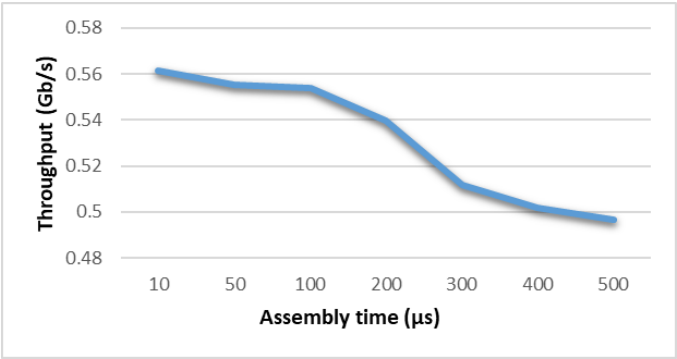

Fig. 5 – *The throughput decreases as the assembly time increases.*

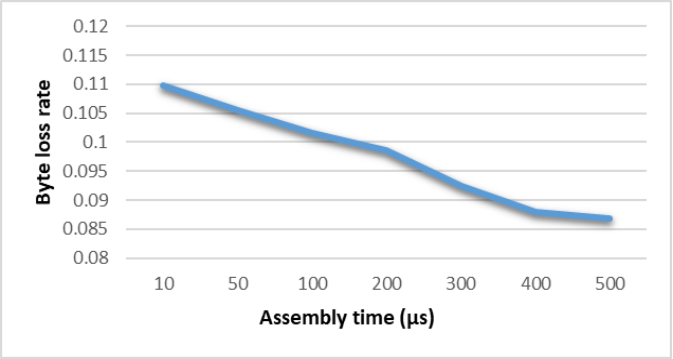

Fig. 6 – *The byte loss rate decreases as the assembly time increases.*

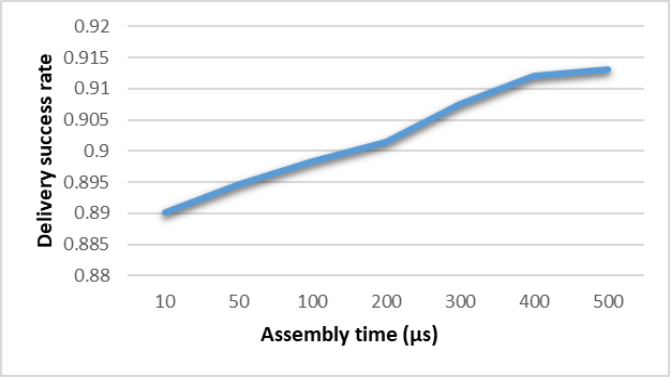

Fig. 7 – *The delivery success rate increases as the assembly time increases.*

When the OBADR method is applied, there is a significant improvement in the throughput when there is a reduction in the buffering time. Specifically, OBADR-based buffering time reduction has reduced TCP Vegas' false congestion detection rate, so it maintains a throughput higher than that when the basic hybrid assembly technique is used (Fig. 8).

However, when investigating the cases with different timers (from 100 µs to 500 µs), it is clear that the best performance is only achieved when the timer is around 300 µs; this timer provides the best byte loss rate (Fig. 9) and the best delivery success rate (Fig. 10). As the timer is longer, this increases the number of packets with an *RTT* exceeding RTT_{max}, so TCP Vegas reduces the congestion window.

The throughput fluctuations extracted for 200 s show that there is a constant adjustment of the congestion window at the Vegas TCP layer, but the throughput achieved by OBADR-based buffering time reduction is always higher than that of the basic hybrid assembly model (Fig. 11).

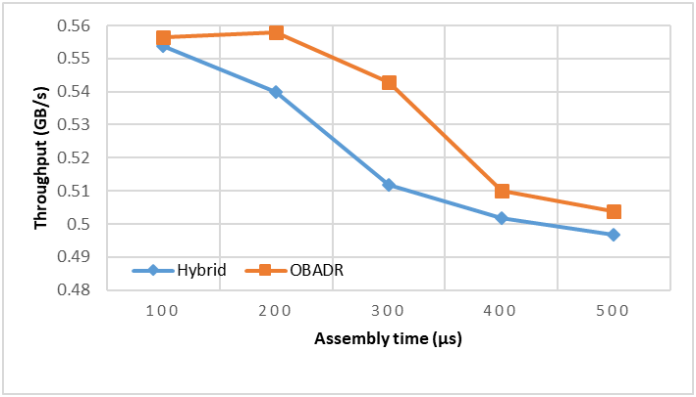

Fig. 8 – *The throughput of OBADR-based buffering time reduction is always higher than that of the hybrid assembly technique for various assembly times.*

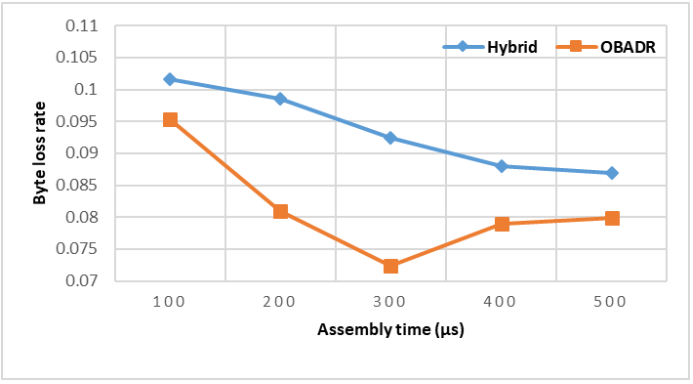

Fig. 9 – *The byte loss rate of OBADR-based buffering time reduction is lower than that of the hybrid assembly technique for various assembly times.*

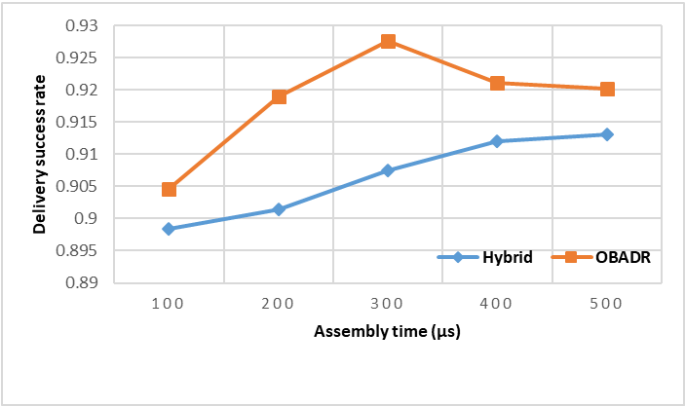

Fig. 10 – *The byte loss rate of OBADR-based buffering time reduction is higher than that of the hybrid assembly technique for various assembly times.*

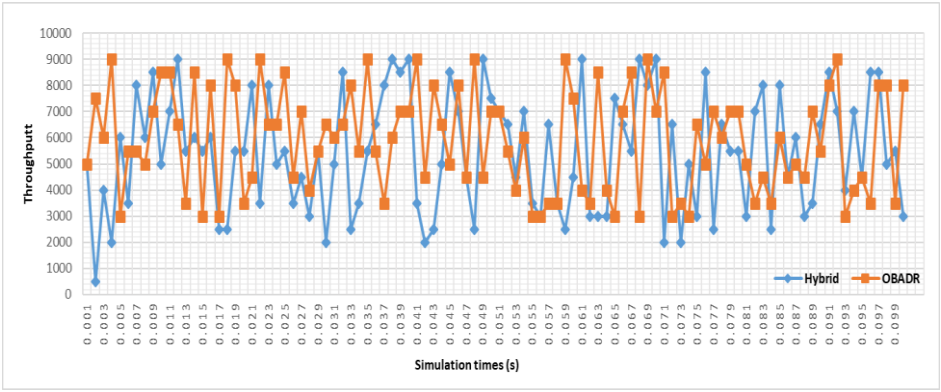

Fig. 11 – *The OBADR-based buffering time reduction always maintains a throughput higher than that of the basic hybrid assembly model.*

6 Conclusion

The all-optical Internet will be a trend in the future, and the combination of TCP and OBS is a potential direction for the all-optical Internet. However, one of the key issues with this combination is false congestion detection due to the long delays in OBS layer operations. To gain a better understanding of this problem, this paper investigates reducing the buffering time at the ingress node to minimise the transmission delay through the OBS network and thereby mitigate false congestion detection in TCP Vegas. The simulation results and analysis demonstrate that reducing the buffering time using the OBADR algorithm is more effective compared to the basic hybrid assembly model. This performance is demonstrated by the system's ability to maintain a high throughput, as the TCP Vegas window shrinkage frequency decreases, while other performance metrics, such as the byte loss rate and successful delivery rate, reach high levels. However, this paper only considers reducing the buffering time at the ingress node and does not address other activities such as retransmission or routing redirection. A latency reduction solution for all OBS layer operations or the combined control of both the TCP and OBS layers could potentially offer a better performance and may represent a new development direction in the future.

7 References

[1] H. Liu, C. F. Lam, C. Johnson: Scaling Optical Interconnects in Datacenter Networks, Opportunities and Challenges for WDM, Proceedings of the 18th IEEE Symposium on High Performance Interconnects, Mountain View, USA, August 2010, pp. 113 – 116.

[2] V. K. Ajay Kumar, K. S. Reddy, A. Prashanth, B. Ravi Kumar: Robust Strategy for Optical Burst Scheduling in WDM Networks Through Void Filling and Segmentation, Proceedings

of the 4th International Conference on Communication and Computational Technologies, Jaipur, India, February 2022, pp. 115 – 129.

[3] S. J. Ben Yoo: Optical Packet and Burst Switching Technologies for the Future Photonic Internet, Journal of Lightwave Technology, Vol. 24, No. 12, December 2006, pp. 4468 – 4492.

[4] A. Zalesky: To Burst or Circuit Switch?, IEEE/ACM Transactions on Networking, Vol. 17, No. 1, February 2009, pp. 305 – 318.

[5] S. Singh, D. Kaur, S. Verma: A Circuit Switching-Based Resource Reservation Approach for Optical Burst Switched Networks, Proceedings of the Third Doctoral Symposium on Computational Intelligence, Lucknow, India, March 2022, pp. 685 – 699.

[6] H. Liu, S. Jiang: A Mixed-Length and Time Threshold Burst Assembly Algorithm Based on Traffic Prediction in OBS Network, International Journal of Sensing, Computing and Control, Vol. 2, No. 2, 2012, pp. 87 – 93.

[7] T. Hashiguchi, X. Wang, H. Morikawa, T. Aoyama: Burst Assembly Mechanism with Delay Reduction for OBS Networks, Proceedings of the Conference on the Optical Internet and 28th Australian Conference on Optical Fibre Technology (COIN/ACOFT), Melbourne, Australia, July 2003, pp. 664 – 666.

[8] Z. Sui, Q. Zeng, S. Xiao: An Offset Differential Assembly Method at the Edge of OBS Network, Proceedings of SPIE – The International Society for Optical Engineering, Vol. 6021, No. 2, December 2005, pp. 1 – 6.

[9] T. Mikoshi, T. Takenaka: Improvement of Burst Transmission Delay Using Offset Time for Burst Assembly in Optical Burt Switching, Proceedings of the 7th Asia-Pacific Symposium on Information and Telecommunication Technologies, Bandos Island, Maldives, April 2008, pp. 13 – 18.

[10] V. M. Nhat Vo, V. Hoa Le, H. S. Nguyen: A Model of Optimal Burst Assembly for Delay Reduction at Ingress OBS Nodes, Turkish Journal of Electrical Engineering and Computer Sciences, Vol. 25, No. 5, October 2017, pp. 3970 – 3982.

[11] V. Jacobson: Congestion Avoidance and Control, ACM SIGCOMM Computer Communication Review, Vol. 18, No. 4, August 1988, pp. 314 – 329.

[12] L. S. Brakmo, S. W. O'Malley, L. L. Peterson: TCP Vegas: New Techniques for Congestion Detection and Avoidance, ACM SIGCOMM Computer Communication Review, Vol. 24, No. 4, October 1994, pp. 24 – 35.

[13] X. Yu, Y. Chen, C. Qiao: A Study of Traffic Statistics of Assembled Burst Traffic in Optical Burst-Switched Networks, Proceedings of the Convergence of Information Technologies and Communications, Boston, USA, July 2002, pp. 149 – 159.

[14] V. M. Nhat Vo, V. Hoa Le, H. S. Nguyen, M. Thanh Le: A Model of QoS Differentiation Burst Assembly with Padding for Improving the Performance of OBS Networks, Turkish Journal of Electrical Engineering and Computer Sciences, Vol. 26, No. 4, July 2018, pp. 1783 – 1795.

[15] O. León, S. Sallent: Issues in TCP Vegas over Optical Burst Switching Networks, Proceedings of the 32nd IEEE Conference on Local Computer Networks (LCN), Dublin, Ireland, October 2007, pp. 242 – 243.

[16] K. Ramantas, K. Vlachos, O. González de Dios, C. Raffaelli: Window Based Burst Assembly Scheme for TCP Traffic Over OBS, Journal of Optical Networking, Vol. 7, No. 5, May 2008, pp. 487 – 495.

[17] J. Sing, B. Soh: TCP New Vegas: Improving the Performance of TCP Vegas Over High Latency Links, Proceedings of the Fourth IEEE International Symposium on Network Computing and Applications, Cambridge, USA, July 2005, pp. 73 – 82.

[18] C. Jin, D. X. Wei, S. H. Low: FAST TCP: Motivation, Architecture, Algorithms, Performance, Proceedings of the IEEE INFOCOM, Hong Kong, China, March 2004, pp. 2490 – 2501.

[19] R. Poorzare: TM Algorithm to Improve Performance of Optical Burst Switching (OBS) Networks, International Journal of Research in Computer Applications and Robotics, Vol. 3, No. 4, April 2015, pp. 45 – 50.

[20] X. Jiang, N. Zhu, L. Yuan: A Novel Burst Assembly Algorithm for OBS Networks Based on Burst Size and Assembly Time Prediction, Journal of Computational Information Systems, Vol. 9, No. 2, January 2013, pp. 463 – 475.

[21] B. Shihada, Q. Zhang, P.- H. Ho: Threshold-Based TCP Vegas Over Optical Burst Switched Networks, Proceedings of the 15th International Conference on Computer Communications and Networks, Arlington, USA, October 2006, pp. 119 – 124.

[22] B. Shihada, Q. Zhang, P.- H. Ho, J. P. Jue: A Novel Implementation of TCP Vegas for Optical Burst Switched Networks, Optical Switching and Networking, Vol. 7, No. 3, July 2010, pp. 115 – 126.

[23] R. Poorzare, S. Jamali: Optimizing TCP Vegas for Optical Networks: A Fuzzy Logic Approach, International Journal of Computer Science and Information Security, Vol. 13, No. 4, April 2015, pp. 33 – 45.

[24] R. Poorzare, S. Abedidarabad: A Novel Implementation of TCP Vegas by Using a Fuzzy-Threshold Base Algorithm to Improve Performance of Optical Networks, Journal of Optical Communications, Vol. 43, No. 2, December 2018, pp. 241 – 249.

[25] S. Abedidarabad, R. Poorzare: Improving Performance of Optical Networks by a Probable Approach, Journal of Optical Communications, Vol. 43, No. 2, January 2019, pp. 251 – 256.

[26] R. Poorzare, A. Calveras, S. Abedidarabad: An Improvement Over TCP Vegas to Enhance its Performance in Optical Burst Switching Networks, Optical Review, Vol. 28, No. 2, April 2021, pp. 215 – 226.

[27] The Network Simulator - NS-2, Available at: https://www.isi.edu/nsnam/ns/.